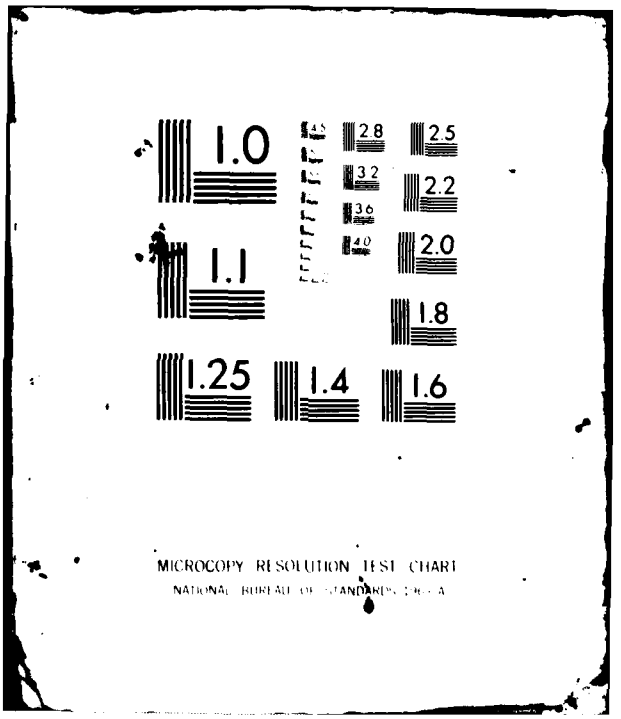
AD-A108 592 ALABAMA UNIV IN HUNTSVILLE SCHOOL OF SCIENCE AND ENG--ETC F/G 16/2
IMPLEMENTATION OF AN OVERLAID VERSION OF PII SIMULATION.(U)
JUL 81 S MANN, D COBB DAAH01-81-D-A006

UNCLASSIFIED DRSMI/RD-81-18-TR NL

END
DATE FILMED
1 82
DTIC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

(2) LEVEL II

AD A108592

TECHNICAL REPORT RD-81-18

IMPLEMENTATION OF AN OVERLAID VERSION OF PII SIMULATION

David Cobb
Systems Simulation
and Development Directorate
US Army Missile Laboratory

Steven Mann
School of Science and Engineering
The University of Alabama, Huntsville
Huntsville, AL

July 1981

U.S. ARMY MISSILE COMMAND
Redstone Arsenal, Alabama 35809

DISPOSITION INSTRUCTIONS

DESTROY THIS REPORT WHEN IT IS NO LONGER NEEDED. DO NOT RETURN IT TO THE ORIGINATOR.

DISCLAIMER

THE FINDINGS IN THIS REPORT ARE NOT TO BE CONSTRUED AS AN OFFICIAL DEPARTMENT OF THE ARMY POSITION UNLESS SO DESIGNATED BY OTHER AUTHORIZED DOCUMENTS.

TRADE NAMES

USE OF TRADE NAMES OR MANUFACTURERS IN THIS REPORT DOES NOT CONSTITUTE AN OFFICIAL INDORSEMENT OR APPROVAL OF THE USE OF SUCH COMMERCIAL HARDWARE OR SOFTWARE.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: RD-81-18	2. GOVT ACCESSION NO.: AD-A108 592
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): IMPLEMENTATION OF AN OVERLAID VERSION OF THE PII SIMULATION	5. TYPE OF REPORT & PERIOD COVERED: Technical Report
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Steven Mann, David Cobb	8. CONTRACT OR GRANT NUMBER(s): DAAH-01-81-D-A006 Delivery Order #0001
9. PERFORMING ORGANIZATION NAME AND ADDRESS: School of Science and Engineering, The University of Alabama in Huntsville, Huntsville, AL 35899	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: US Army Missile Command, ATTN: DRSMI-RPT, Redstone Arsenal, AL 35899	12. REPORT DATE: July 1981
	13. NUMBER OF PAGES: 17
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

PERSHING II
Digital Trajectory Simulation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

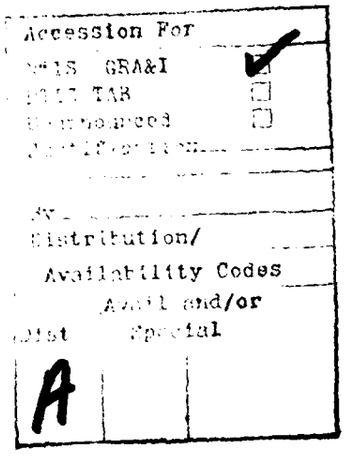
The techniques for implementing a very large PERSHING II digital simulation on a Perkin Elmer minicomputer are discussed in this report. The simulation was broken into subunits and overlaid. A summary is given on the software requirements for operating the simulation.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

TABLE OF CONTENTS



I. INTRODUCTION

The role of the University of Alabama in Huntsville (UAH) in this project has been to provide support on the two Perkin Elmer minicomputers available. This specifically involved setting up a working version of the PII digital simulation (U70), and at the same time permitting rapid, easy modification.

The major problem advanced was that the simulation was very large, much too large to run using normal methods. Therefore, the program was broken into subunits and was overlaid. This presented a multiplicity of problems which will be discussed. The software for U70 operations is presented in detail.

II. THE OVERLAY STRUCTURE

U70 was delivered to the Systems Evaluation Branch as a large block of FORTRAN code. This indicated that the program was executed in the normal method, which required all executable codes to be in memory at the issuance of the start command. This required approximately 900K Bytes of memory. The constraint for Perkin Elmer use was 330K Bytes, so obviously overlaying was in order. Even at 330KB and U70 user was the only possible user because all memory was absorbed by the U70 task.

Overlaying is a technique which permits discs to act as psuedo main memory. Using this method only an overlay containing a called subprogram is loaded in main memory. The overlay may contain one or more subprograms. Figure 1 illustrates an overlaid structure.

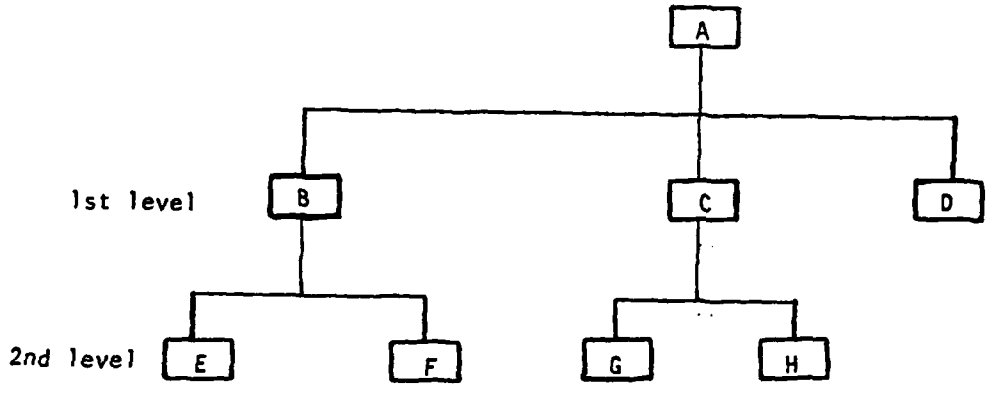


Figure 1. Overlaid structure.

At any time only the subprograms in the root overlay, any one of the first level and any one of the second level, can be in memory. For example, only A, B, and E can be in memory at a time. Therefore, subprograms in E cannot call routines in C, D, G, or H. The total size of the loaded task (the allocated size of memory to be used) is the total of the root overlay, the largest overlay in the first level and the largest overlay in the second level. The root overlay contains routines which are often used or are in more than one overlay. Using this method the size of the task is approximately 310K Bytes.

III. COMPILING U70

U70 is compiled using the Development (D) compiler with the same CSS commands as usual except that logical unit (lu) 9 is assigned to a file (U70TIM. COM) containing the major common blocks used by the U70 subroutines. The file delivered to Systems Evaluation Branch was composed of a single file which contained all U70 routines. For compilation ease this file was broken into multiple files, most of which contain only one subprogram. This permits a change to be made quickly, since only one subroutine must be recompiled rather than all the subroutines contained in one file. If the size of the file to be compiled is large, it is often necessary to increase the load size of the compiler. This is needed to stop the compiler from putting part of the symbol table out on disc rather than maintaining it in memory. This problem causes the compiler to be extremely slow. One hundred K Bytes is enough to keep the symbol table from paging to disc. The CSS command used to compile a U70 program unit is U70C. An FTN (FORTRAN) extension is assumed on the file to be compiled.

U70C @1, @2, @3, @4, @5

@1: Input source file

@2: Object filename, defaults to @1.OBJ

@3: List filename, defaults to @1.LST

@4: Start Option(s) separated with spaces, defaults to no options

@5: Load size for F7D, d-faults to 50K Bytes.

The user also has the option of compiling all PII source files by entering the U70CA command. This will execute the CSS file U70CA. No input parameters are required. U70CA deletes all List (LST) files, therefore, if a LST file is required for debugging the subprogram should be recompiled using U70C. If an error occurs in U70CA, the file being compiled at the time of the error will not be compiled. Figure 2 list U70 source files.

IV. LINKING U70

Linking is the process by which calls to externals are resolved and an executable image of the object is produced. In the case of U70, the executable task is on overlaid task, with the overlays defined in the linking process.

The overlays are structured such that no subprogram calls to any other subprogram in another overlay. An overlay must be loaded only once. In developing the overlay structure, a level of 2 was used, but this caused data communication problems between subprogram units contained in different overlays. Under all circumstances the data assigned to all variables and arrays not contained in a common block or passed was lost on execution of an END command. Using a SAVE command the data could be saved, but only until the overlay was reloaded. The communication problem necessitated the use of only level one overlays. Using level one overlaying, a unit only needed to be loaded once,

```
ARD59D.FTN
BEE   .FTN
CONV
FASTD
FASTC
SORTIT
SPEEDY
WRTAPE
BINIT
BGUID
FUNC
VACRAN
FOLD
GRAPH
NINIT
ENPUT
RVEXEC
STAT
RVOUT
RVGUID
QUAD
RVINIT
RVAPLT
RVAERO
SCS
GAUSS
RANDU
BEXEC
THRUST
TRIMA
BOUT
BAPLT
RATLIM
BAERO
```

Figure 2. U70 source files.

which permitted variables to be saved with the inclusion of a SAVE command. This increased the task size but made the simulation operational.

Another previously unknown problem encountered was that an object or objects produced by an R04.0 compiler cannot be linked by TET; if the task is complex, LINK must be used. U70 is linked using the command U70LINK. This command executes the instruction contained in U70LINK.CSS. No input parameters are required.

V. RUNNING U70

U70 is executed using the commands UST1 and UST2. These commands invoke CSS files containing the commands to load the task, make logical unit assignments, and start the task. UST1 is used to create a file assigned to LU 11 which is used as the major input source for U70 when UST2 is called.

UST1 is called as a preparatory step in executing U70. It is only necessary to use UST1 occasionally; therefore, it is maintained as a separate file. Both UST1 and UST2 expect a */* as an end of data marker; therefore, all input files must end in a */*.

VI. DEBUGGING U70

U70 can be debugged using the normal methods, but because of the size and complexity of U70 the use of a task map is a necessity. Task maps contain all the addresses of the subprogram units in a FORTRAN program. Using the map in conjunction with the FORTRAN list (LST) file and the error address, the line in error can be pinpointed. This allows a user to find a line in error out of several thousand.

The first step in finding the error address is to compare the address given during program execution with the program (P) addresses in the map. All program addresses greater than the execution address are excluded. The address just less than the error address defines the subprogram which contains the error. This program address is then subtracted from the execution error address. The result is the offset into the subprogram which contains the problem. The offset points to a line of FORTRAN code, which can be found using the LST file of the subprogram which contains the problem.

APPENDIX A

PROGRAM LISTING

```
*
*      U70C.CSS
*   @1 = INPUT FILE
*   @2 = OBJECT FILE (DEFAULT @1.OBJ)
*   @3 = LIST FILE (DEFAULT PR:)
*   @4 = START OPTIONS
*   @5 = LOAD SIZE FOR F7D (DEFAULT 20K)
*
$IFNU @1;$COPY
**@0: MISSING FIRST PARAMETER
$NOC
$CLEAR;$ENDC
$IFNX @1.FTN;$COPY
**@0: INPUT FILE NON-EXISTANT
$NOC
$CLEAR;$ENDC
$IFNU @5;LO .BG,F7D,3C;$ENDC
$IFNN @5;LO .BG,F7D,@5;$ENDC
T .BG
AS 1,@1.FTN,SRO
$IFNN @2
     ALLONEW @2.OBJ,126/2
     AS 2,@2.OBJ
$ENDC
$IFNU @2
     ALLONEW @1.OBJ,126/2
     AS 2,@1.OBJ
$ENDC
ALLONEW @1.LST
$IFNU @3;AS 3,@1.LST;$ENDC
$IFNN @3;$IFNX @3;$COPY
**@0: LIST FILE OR DEVICE NON-EXISTANT
$NOC;$CLEAR;$ENDC
     AS 3,@3,EW0;$ENDC
AS 9,U70TIM.CCM
ST ,@4
$IFG 3
      $IFE 4;$COPY
**@0: COMPILATION ERRORS !!!
$NOC
      $ENDC
      $IFE 6;$COPY
**@0: BAD START OPTIONS - @4
$NOC
      $ENDC
      $IFE 8;$COPY
**@0: MEMORY OVERFLOW - INCREASE LOAD SIZE: @5
$NOC
      $ENDC
      $IFE 10;$COPY
**@0: ECM/EOF ON SYMBOL TABLE FILE - DELETE OLD FILES
$NOC
      $ENDC
$ENDC
$EXIT
```

```
*      U70CA.CSS
* USED TO COMPILE ALL PII FILES
*
$JOB
U70C U70MAIN,,,,100
STERMJOB
*
$J
U70C ARD59D,,,,10C
ST
*
$J
U70C BEE,,,,1C0
ST
*
$J
U70C CONV,,,,100
ST
*
$J
U70C FASTD,,,,100
ST
*
$J
U70C FASTC,,,,100
ST
*
$J
U70C SORTIT,,,,10C
ST
*
$J
U70C SPEEDY,,,,10C
ST
*
$J
U70C WRTAPE,,,,100
ST
*
$J
U70C BINIT,,,,100
ST
*
$J
U70C BGUID ,,,,100
ST
*
$J
U70C FUNC,,,,100
ST
*
$J
U70C VACRAN,,,,100
ST
*
$J
U70C FOLD,,,,100
ST
*
$J
U70C GRAPH,,,,100
ST
```

U70CA.CSS (Cont'd)

```
*
$J
U70C NINIT,,,,100
$T
*
$J
U70C ENPUT,,,,100
$T
*
$J
U70C RVEXEC,,,,100
$T
*
$J
U70C STAT,,,,100
$T
*
$J
U70C RVCUT,,,,100
$T
*
$J
U70C RVGUID,,,,100
$T
*
$J
U70C QUAD,,,,100
$T
*
$J
U70C RVINIT,,,,100
$T
*
$J
U70C RVAPLT,,,,100
$T
*
$J
U70C RVAERO,,,,100
$T
*
$J
U70C SCS,,,,100
$T
*
$J
U70C RANDU,,,,100
$T
*
```

```
$J
U70C GAUSS,,,,100
$T
*
$J
U70C BEXEC,,,,100
$T
*
$J
U70C THRUST,,,,100
$T
*
$J
U70C TRIMA,,,,1003
$T
*
$J
U70C BOUT,,,,100
$T
*
$J
U70C BAPLT,,,,100
$T
*
$J
U70C RATLIM,,,,100
$T
*
$J
U70C BAERO,,,,100
$T
*
```

U70CA.CSS (Cont'd)

```
XDE U70MAIN.LST
XDE ARD59D.LST
XDE BEE.LST
XDE CONV.LST
XDE FASTD.LST
XDE FASTC.LST
XDE SCRTIT.LST
XDE SPEEDY.LST
XDE WRTAPE.LST
XDE BINIT.LST
XDE BGUID.LST
XDE FUNC.LST
XDE VACRAN.LST
XDE FOLD.LST
XDE GRAPH.LST
XDE NINIT.LST
XDE ENPUT.LST
XDE STAT.LST
XDE RVOUT.LST
XDE RVGUID.LST
XDE QUAD.LST
XDE RVINIT.LST
XDE RVAPLT.LST
XDE RVAERO.LST
XDE SCS.LST
XDE GAUSS.LST
XDE RANDU.LST
XDE BEXEC.LST
XDE THRUST.LST
XDE TRIMA.LST
XDE BCUT.LST
XDE BAPLT.LST
XDE RATLIM.LST
XDE BAERO.LST
$CLEAR
$EXIT
```

```
*      U7OLINK.CSS
L .BG,LINK
T .BG
XDE U7OLINK.MAP
AL U7OLINK.MAP,IN
XDE LINK.CMD
XDE U7O.TSK
*
$BUILD LINK.CMD
ESTABLISH TASK
OP DF
OP FL
OP WO=1500
OP NAPPAUSE
OP IOBLOCKS=10
OP SYSSPACE=5000
MAP U7OLINK.MAP,AD,XR
*
*
INCL U7OMAIN
INCL ARD59D
INCL BEE
INCL CONV
INCL FASTD
INCL FASTC
INCL SORTIT
INCL SPEEDY
INCL WRTAPE
INCL BINIT
INCL BGUID
INCL FUNC
INCL VACRAN
INCL FOLD
INCL GRAPH
*
*
OVERLAY ONE,1
INCLUDE NINIT
*
*
OVERLAY TWO,1
INCL ENPUT
*
```

U70LINK.CSS (Cont'd)

```
*
OVERLAY THREE,1
INCLUDE RVEXEC
INCLUDE STAT
INCL RVOUT
INCL RVGUID
INCL GUAD
INCL RVINIT
INCL RVAPLT
INCL RVAERO
INCL SCS
INCL GAUSS
INCL RANDU
*
*
*
OVERLAY FOUR,1
INCLUDE BEXEC
INCLUDE THRUST
INCL TRIMA
INCL BOUT
INCL BAPLT
INCL RATLIM
INCL BAERO
*
*
*
LIBRARY F7RTL.OBJ/S
BUILD U70.TSK
END
$END3
ST /C=LINK.CMD/L=NULL:
$EXIT
```

UST1. CSS

```
*
L .BG,U7O
T .BG
XDE TAPE11.DAT
XDE U7OIN1.DAT
AL TAPE11.DAT,IN
AL U7OIN1.DAT,IN
AS 0,CON:
AS 1,NULL:
AS 2,CON:
AS 3,PR:
AS 4,U7CIN1.DAT
AS 5,U7OINPUT.DAT
AS 6,PR:
AS 7,NULL:
AS 8,CON:
AS 9,NULL:
AS 10,NULL:
AS 11,TAPE11.DAT
AS 12,NULL:
AS 13,NULL:
AS 14,NULL:
ST
$EXIT
```

UST2. CSS

```
*
L .BG,U70
T .BG
XDE NINE.DAT
AL NINE.DAT,TN
AS 0,CON:
AS 1,NULL:
AS 2,CON:
AS 3,PR:
AS 4,U7CIN1.DAT
AS 5,COB32.DAT
AS 6,PR:
AS 7,NULL:
AS 8,CON:
AS 9,NINE.DAT
AS 10,NULL:
AS 11,TAPE11.DAT
AS 12,NULL:
AS 13,NULL:
AS 14,NULL:
ST
$EXIT
```

DISTRIBUTION

	No. of Copies
Defense Technical Information Center Cameron Station Alexanderia, VA 22314	2
US Army Materiel Systems Analysis Activity ATTN: DRXSY-MP Aberdeen Proving Ground, MD 21005	1
IIT Research Institute ATTN: GACIAC 10 West 35th Street Chicago, IL 60616	1
School of Science and Engineering The University of Alabama, Huntsville ATTN: Steven Mann Huntsville, AL 35807	1
DRSMI-R, Dr. McCorkle	1
-RD, Dr. Grider	1
-RDF, Mr. D. Cobb	6
-LP, Mr. Voigt	1
-RPR	3
-RPT (Reference Copy)	1
(Record Copy)	1